



Automata and temporal logic

— *a biased overview* —

Stephan Merz

<http://www.loria.fr/~merz/>

INRIA Lorraine & LORIA, Nancy



Plan of this talk



- from tableaux to Büchi automata
- alternating automata for LTL
- branching-time logics



Plan of this talk



- from tableaux to Büchi automata

a crash course in pre-history

- alternating automata for LTL

some innovations since the 1990s

- branching-time logics

nice theory, but does it work?



Tableaux for modal logic



- decomposition of formulas : top-down

- closure criteria : bottom-up, determine satisfiability



Tableaux for modal logic



- decomposition of formulas : top-down

$$\frac{\Gamma, \varphi \wedge \psi}{\Gamma, \varphi, \psi} \quad \frac{\Gamma, \varphi \vee \psi}{\Gamma, \varphi} \quad \frac{l_1, \dots, l_k, \Box \varphi_1, \dots, \Box \varphi_m, \Diamond \psi_1, \dots, \Diamond \psi_n}{\varphi_1, \dots, \varphi_m, \psi_1 \quad \dots \quad \varphi_1, \dots, \varphi_m, \psi_n}$$

- closure criteria : bottom-up, determine satisfiability



Tableaux for modal logic



- decomposition of formulas : top-down

$$\frac{\Gamma, \varphi \wedge \psi}{\Gamma, \varphi, \psi} \quad \frac{\Gamma, \varphi \vee \psi}{\Gamma, \varphi} \quad \frac{l_1, \dots, l_k, \Box \varphi_1, \dots, \Box \varphi_m, \Diamond \psi_1, \dots, \Diamond \psi_n}{\varphi_1, \dots, \varphi_m, \psi_1 \quad \dots \quad \varphi_1, \dots, \varphi_m, \psi_n}$$

- rules reflect properties of accessibility relation

$$\frac{\Gamma, \Box A}{\Gamma, A} \quad \Box \text{ reflexive} \quad \frac{\Gamma}{\Gamma, \Diamond \top} \quad \Box \text{ serial}$$

- closure criteria : bottom-up, determine satisfiability



Tableaux for LTL



- decomposition based on recursive laws

$$\mathbf{G} \varphi \leftrightarrow \varphi \wedge \mathbf{XG} \varphi$$

$$\mathbf{F} \varphi \leftrightarrow \varphi \vee \mathbf{XF} \varphi$$



Tableaux for LTL



- decomposition based on recursive laws

$$\mathbf{G} \varphi \leftrightarrow \varphi \wedge \mathbf{XG} \varphi$$

$$\mathbf{F} \varphi \leftrightarrow \varphi \vee \mathbf{XF} \varphi$$

- LTL tableaux contain loops

$$\left[\frac{\mathbf{G} p}{p, \mathbf{XG} p} \right]$$

$$\left[\frac{\frac{\mathbf{GF} p}{\mathbf{F} p, \mathbf{XGF} p}}{\mathbf{XF} p, \mathbf{XGF} p} \mathbf{F} p, \mathbf{GF} p \right]$$



Tableaux for LTL



- decomposition based on recursive laws

$$\mathbf{G} \varphi \leftrightarrow \varphi \wedge \mathbf{XG} \varphi$$

$$\mathbf{F} \varphi \leftrightarrow \varphi \vee \mathbf{XF} \varphi$$

- LTL tableaux contain loops



- distinguish “good” from “bad” loops



Tableaux for LTL



- decomposition based on recursive laws

$$G\varphi \leftrightarrow \varphi \wedge XG\varphi$$

$$F\varphi \leftrightarrow \varphi \vee XF\varphi$$

- LTL tableaux contain loops



- distinguish “good” from “bad” loops



ω -automata



- separate concerns: logic + combinatorics

translation

$$\varphi \mapsto \mathcal{A}_\varphi$$

$$\mathcal{L}(\mathcal{A}_\varphi) = \text{models of } \varphi$$

emptiness checking

$$\mathcal{L}(\mathcal{A}) \stackrel{?}{=} \emptyset$$

(efficient, if possible)



ω -automata



- separate concerns: logic + combinatorics

translation $\varphi \mapsto \mathcal{A}_\varphi$ $\mathcal{L}(\mathcal{A}_\varphi) = \text{models of } \varphi$

emptiness checking $\mathcal{L}(\mathcal{A}) \stackrel{?}{=} \emptyset$ (efficient, if possible)

- useful for satisfiability *and* model checking

$\mathcal{T} \models \varphi$ (\mathcal{T} transition system)

iff

every run of \mathcal{T} satisfies φ

iff

$$\mathcal{L}(\mathcal{T} \times \mathcal{A}_{\neg\varphi}) = \emptyset$$



Büchi automata



• $\mathcal{A} = (\Sigma, Q, I, \delta, F)$

Σ alphabet

Q finite set of locations

$I \subseteq Q$ initial locations

$\delta \subseteq Q \times \Sigma \times Q$ transition relation

$F \subseteq Q$ accepting locations

• run of \mathcal{A} over ω -word

$$q_0 \xrightarrow{a_0} q_1 \xrightarrow{a_1} q_2 \xrightarrow{a_2} \dots$$

initialization $q_0 \in I$

consecution $(q_i, a_i, q_{i+1}) \in \delta$

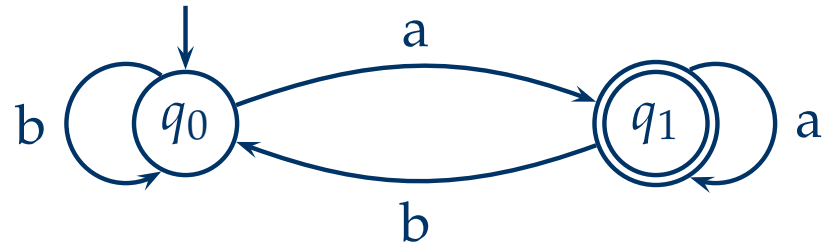
acceptance $q_i \in F$ for infinitely many i



Büchi automata: examples



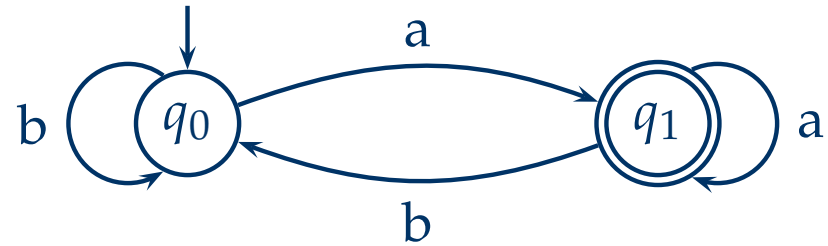
- infinitely often 'a'



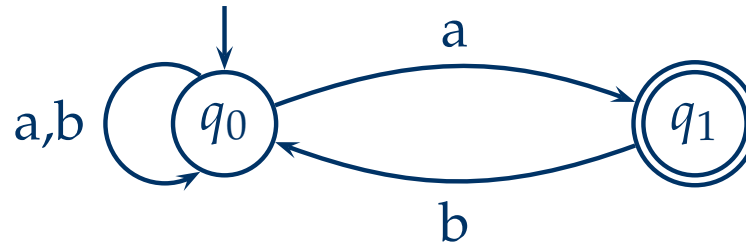
Büchi automata: examples



- infinitely often 'a'



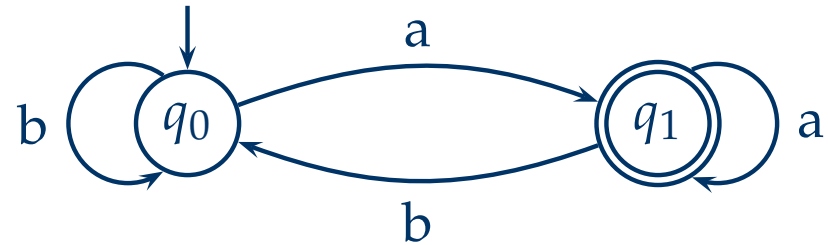
- infinitely often 'ab'



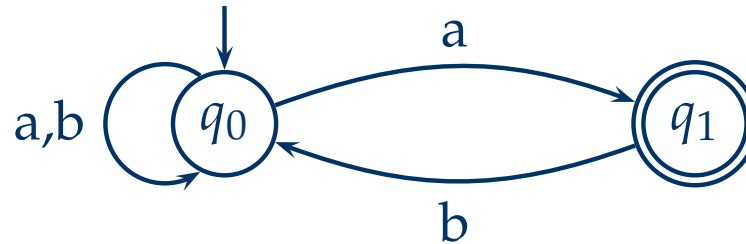
Büchi automata: examples



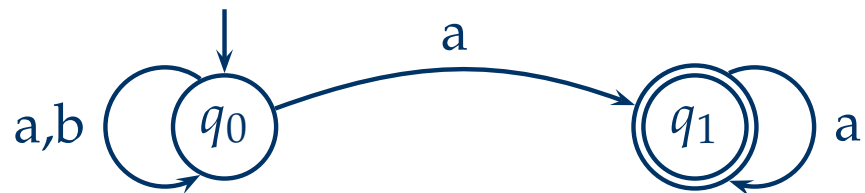
- infinitely often 'a'



- infinitely often 'ab'



- eventually only 'a'



no equivalent deterministic Büchi automaton



Büchi automata: results



- emptiness checking

$$\mathcal{L}(\mathcal{A}) \neq \emptyset \quad \text{iff} \quad q_0 \xrightarrow{\Sigma^*} q \xrightarrow{\Sigma^+} q \quad \text{for some } q_0 \in I, q \in F$$

“ \Leftarrow ” obvious

“ \Rightarrow ” any accepting run contains repetition of some $q \in F$



Büchi automata: results



- emptiness checking

$\mathcal{L}(\mathcal{A}) \neq \emptyset$ iff $q_0 \xrightarrow{\Sigma^*} q \xrightarrow{\Sigma^+} q$ for some $q_0 \in I, q \in F$

“ \Leftarrow ” obvious

“ \Rightarrow ” any accepting run contains repetition of some $q \in F$

\rightsquigarrow decidable in linear time (NLOGSPACE)



Büchi automata: results



- emptiness checking

$\mathcal{L}(\mathcal{A}) \neq \emptyset$ iff $q_0 \xrightarrow{\Sigma^*} q \xrightarrow{\Sigma^+} q$ for some $q_0 \in I, q \in F$

“ \Leftarrow ” obvious

“ \Rightarrow ” any accepting run contains repetition of some $q \in F$

\rightsquigarrow decidable in linear time (NLOGSPACE)

- closure properties

union standard NFA construction

intersection “marked” product

complement difficult (Safra 1988: $O(2^{n \log n})$ locations)



Büchi automata: variants



- Generalized Büchi $\mathcal{A} = (\Sigma, Q, I, \delta, \{F_1, \dots, F_n\})$
 - acceptance infinitely many $q_i \in F_k$, for all k
 - intersection simple product construction
 - translate to Büchi automaton using counter (mod n)



Büchi automata: variants



- Generalized Büchi $\mathcal{A} = (\Sigma, Q, I, \delta, \{F_1, \dots, F_n\})$

acceptance infinitely many $q_i \in F_k$, for all k

intersection simple product construction

translate to Büchi automaton using counter (mod n)

- Muller automaton $\mathcal{A} = (\Sigma, Q, I, \delta, \mathcal{F})$

acceptance set of locations visited infinitely often $\in \mathcal{F}$

Streett automata exponentially more succinct than Büchi



From LTL to GBA



- basic insight
 - let \mathcal{V} be set of atoms containing those of formula φ
 - interpret temporal structure as ω -word over alphabet $2^{\mathcal{V}}$
 - models of φ define ω -language $\mathcal{L}(\varphi)$



From LTL to GBA



- basic insight
 - let \mathcal{V} be set of atoms containing those of formula φ
 - interpret temporal structure as ω -word over alphabet $2^{\mathcal{V}}$
 - models of φ define ω -language $\mathcal{L}(\varphi)$
- construct \mathcal{A}_φ such that $\mathcal{L}(\mathcal{A}_\varphi) = \mathcal{L}(\varphi)$

locations	sets of “subformulas” of φ promised to be true
initial locations	locations promising φ
transition relation	non-temporal formulas: ensure satisfaction temporal formulas: apply recursion laws
accepting locations	defined from “eventualities” $\mathbf{F} \psi, \psi_1 \mathbf{U} \psi_2$



GBA for $\mathbf{GF} p \wedge \mathbf{GF} \neg p$



- construct the tableau for the formula

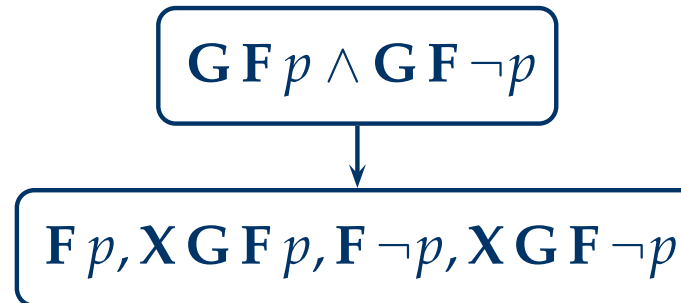
$$\mathbf{GF} p \wedge \mathbf{GF} \neg p$$



GBA for $\mathbf{GF} p \wedge \mathbf{GF} \neg p$



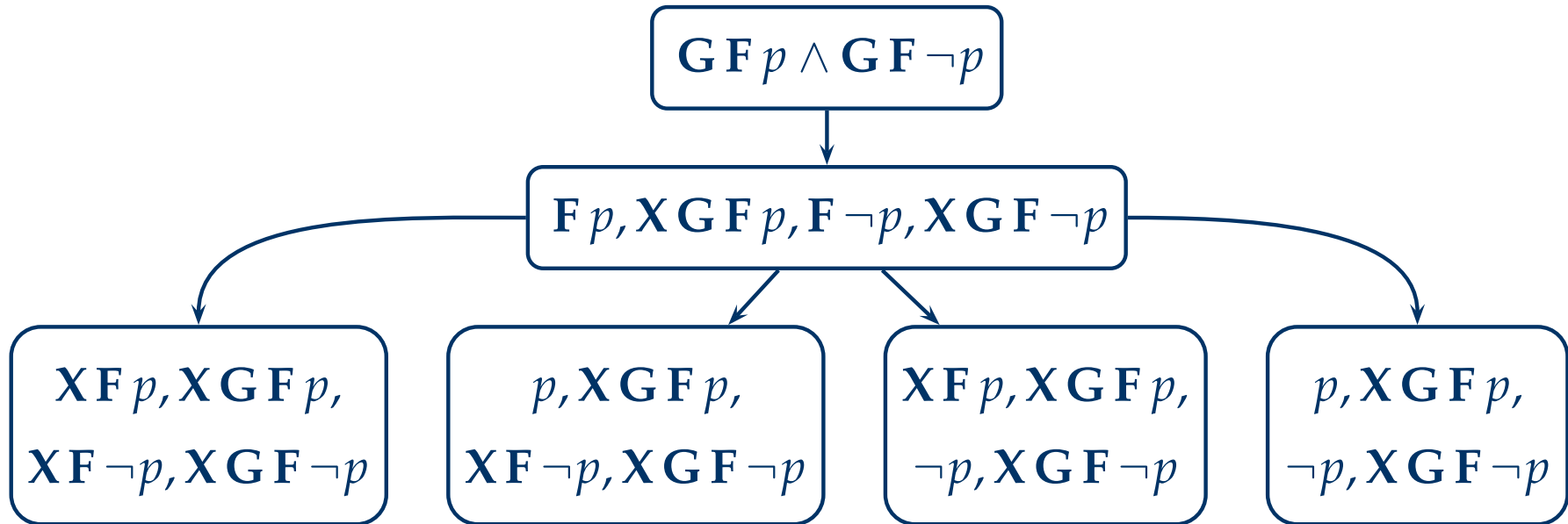
- construct the tableau for the formula



GBA for $\mathbf{GF} p \wedge \mathbf{GF} \neg p$



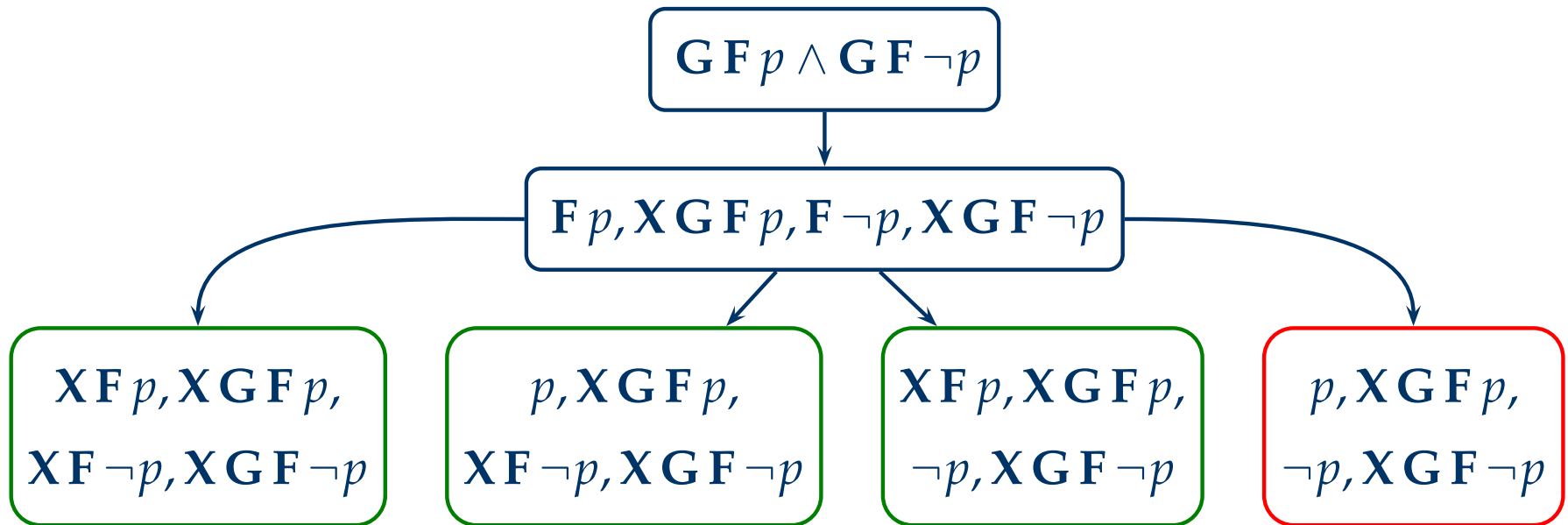
- construct the tableau for the formula



GBA for $\mathbf{GF} p \wedge \mathbf{GF} \neg p$



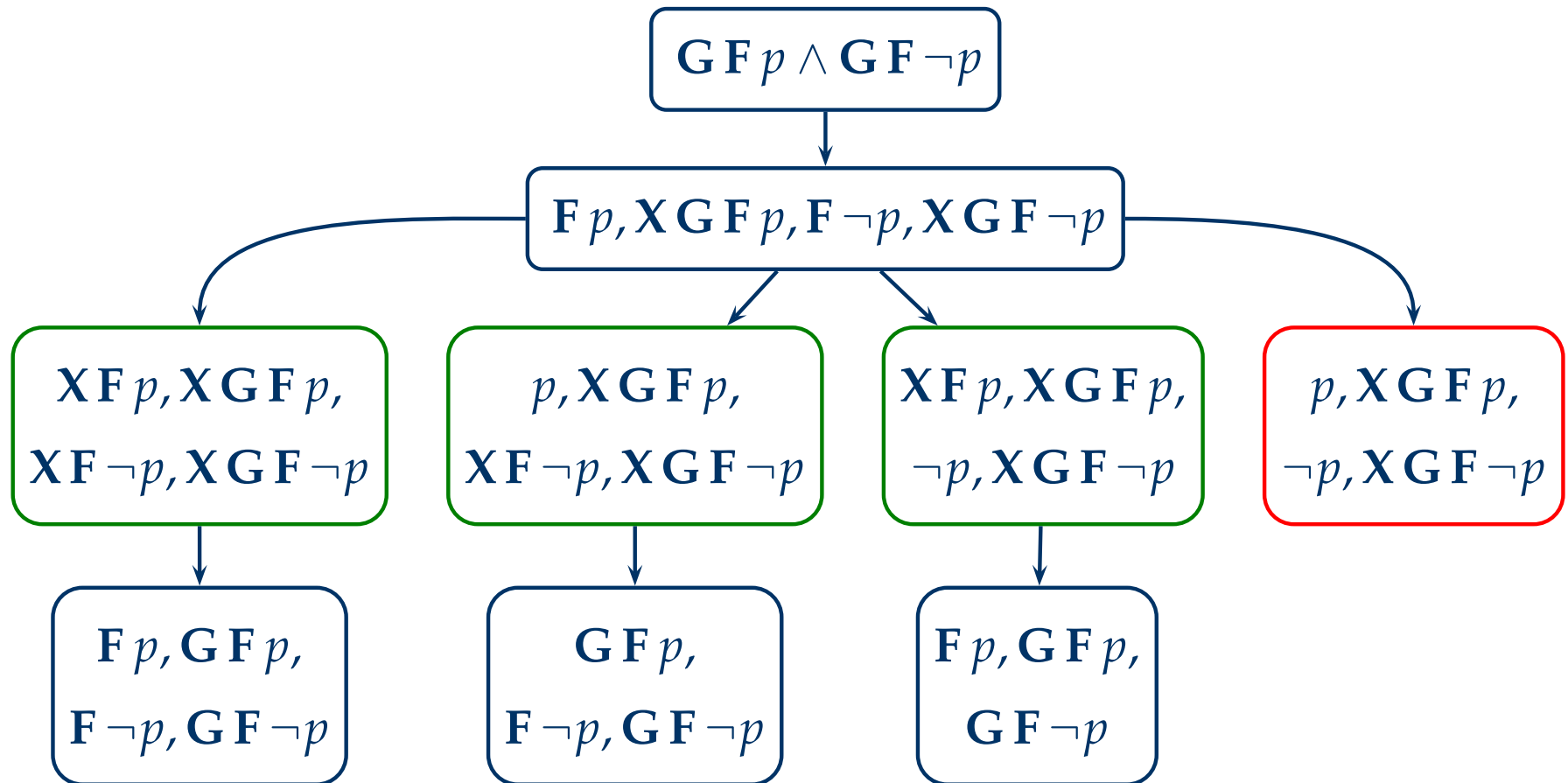
- construct the tableau for the formula



GBA for $\mathbf{GF} p \wedge \mathbf{GF} \neg p$



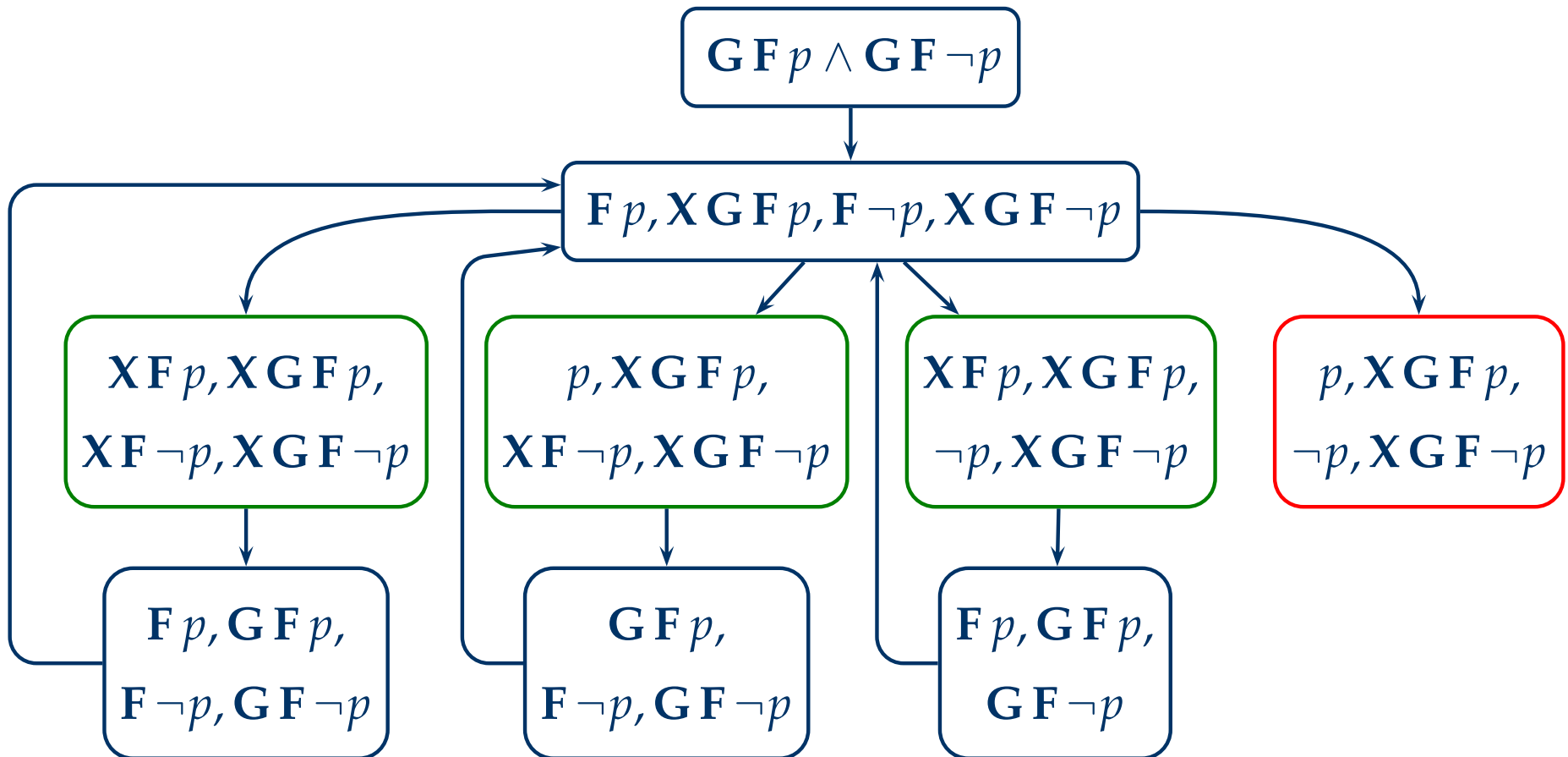
- construct the tableau for the formula



GBA for $\mathbf{GF} p \wedge \mathbf{GF} \neg p$



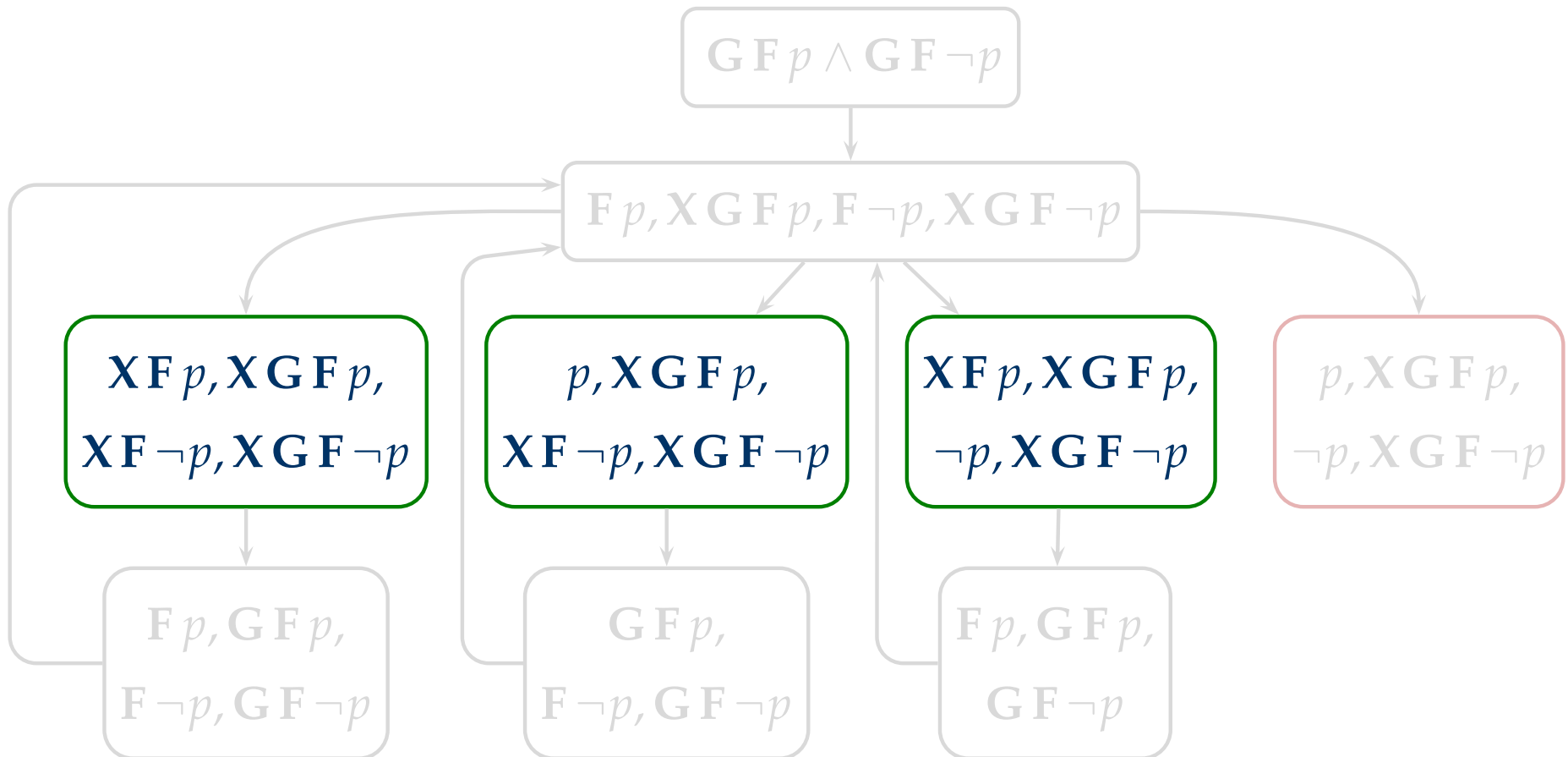
- construct the tableau for the formula



GBA for $\mathbf{GF} p \wedge \mathbf{GF} \neg p$



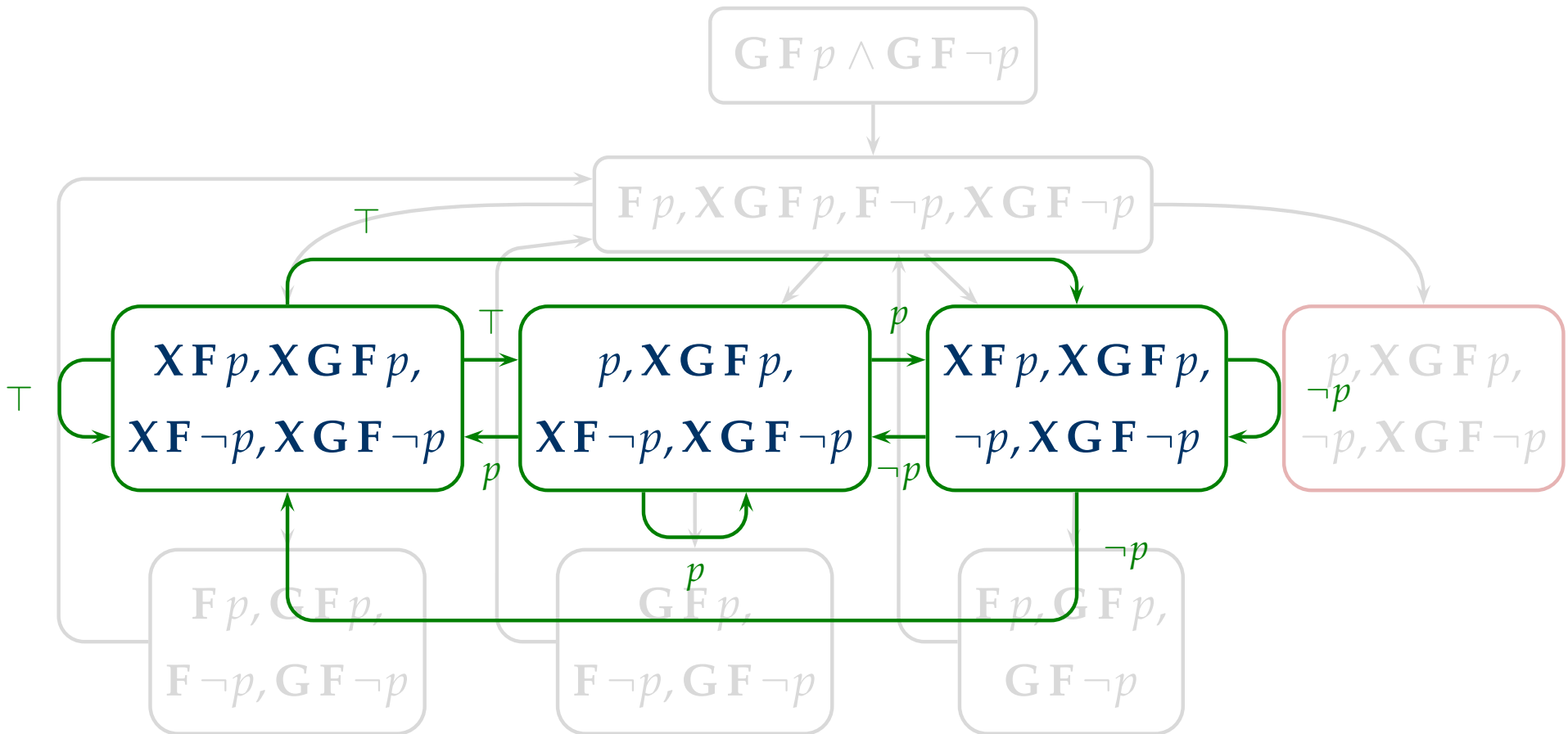
- tableau states yield locations of GBA



GBA for $\mathbf{GF} p \wedge \mathbf{GF} \neg p$



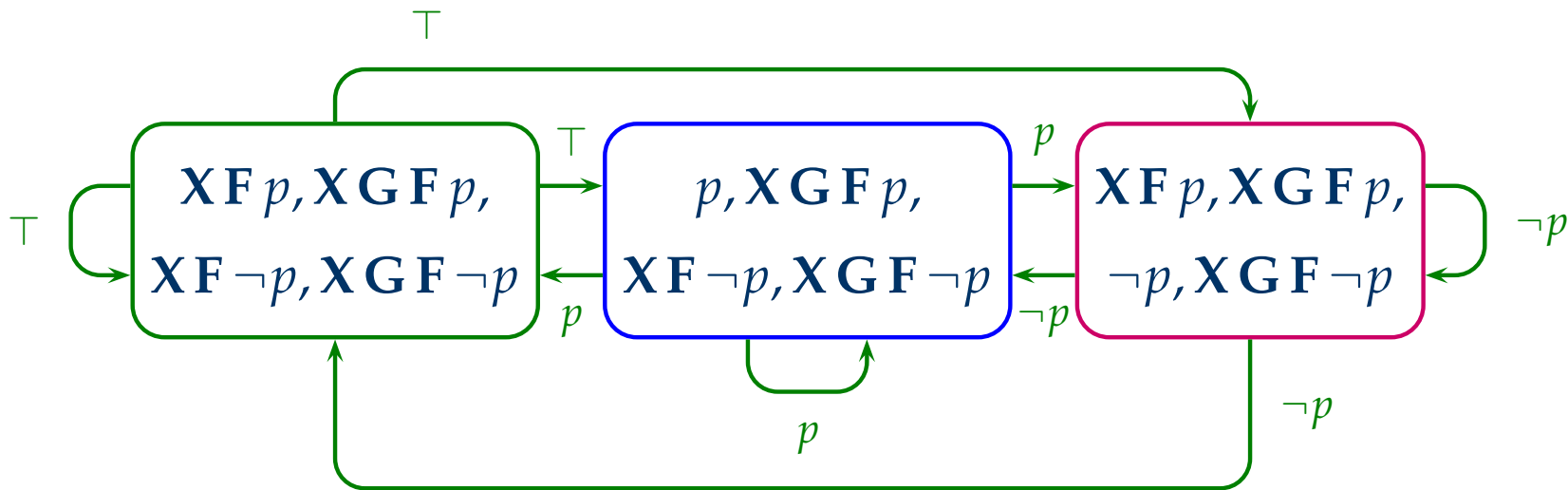
- follow paths in tableau to obtain GBA transitions



GBA for $\mathbf{GF} p \wedge \mathbf{GF} \neg p$



- acceptance conditions from formulas $\mathbf{F} p$ and $\mathbf{F} \neg p$



Practical considerations



- size of \mathcal{A}_φ exponential in length of φ
 - minimizing Büchi automata is hard
 - analyze SCCs to reduce automaton size
[Gastin & Oddoux, 2001]
 - apply simulation relations to reduce automaton size
[Etessami et al, 2000, 2001]
- worst-case complexity of LTL model checking

$$O(|\mathcal{T}| \cdot |\mathcal{A}_{\neg\varphi}|) = O(|\mathcal{T}| \cdot 2^{|\varphi|})$$



State explosion problem



- $\mathcal{T} \times \mathcal{A}_{\neg\varphi}$ is too big to be computed effectively
- problems start around $10^6 - 10^7$ states
- partial remedies (mainly concerning \mathcal{T})

reduce ignore irrelevant parts of $\mathcal{T} \times \mathcal{A}_{\neg\varphi}$

on-the-fly interleave construction and verification

partial-order identify equivalent interleavings

compress construct compact representations of $\mathcal{T} \times \mathcal{A}_{\neg\varphi}$

abstract “coarser” representation of \mathcal{T}



Extensions



- Linear-time logics

PLTL straightforward variant [Lichtenstein&Pnueli, 1984]

NLTL doubly exponential [Laroussinie et al, 2002]

QPTL non-elementary [Sistla et al, 1987]

- Branching-time logics

satisfiability similar theory around tree automata

model checking tree automata introduce exponential blowup



Summing up



- mature theory and efficient implementations
- logic and combinatorics nicely decoupled
- disadvantages
 - somewhat clumsy construction
 - automaton generation can be bottleneck
 - branching-time model checking does not fit into framework



Alternating automata (for LTL)



• $A = (\mathcal{V}, Q, q_0, \delta, \mathcal{F})$

\mathcal{V} finite set of atomic propositions

Q finite set of locations ($\mathcal{V} \cap Q = \emptyset$)

$q_0 \in Q$ initial location

$\delta : Q \rightarrow \mathcal{B}^+(\overline{\mathcal{V}} \cup Q)$ transition function

\mathcal{F} acceptance condition (e.g., Büchi)



Alternating automata (for LTL)



- $A = (\mathcal{V}, Q, q_0, \delta, \mathcal{F})$

\mathcal{V} finite set of atomic propositions

Q finite set of locations ($\mathcal{V} \cap Q = \emptyset$)

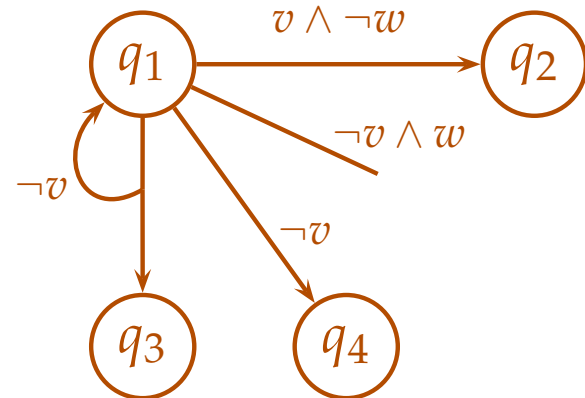
$q_0 \in Q$ initial location

$\delta : Q \rightarrow \mathcal{B}^+(\overline{\mathcal{V}} \cup Q)$ transition function

\mathcal{F} acceptance condition (e.g., Büchi)

- interpretation of transitions

$$\begin{aligned} \delta(q_1) = & v \wedge \neg w \wedge q_2 \\ & \vee \neg v \wedge (q_4 \vee (q_1 \wedge q_3)) \\ & \vee \neg v \wedge w \end{aligned}$$



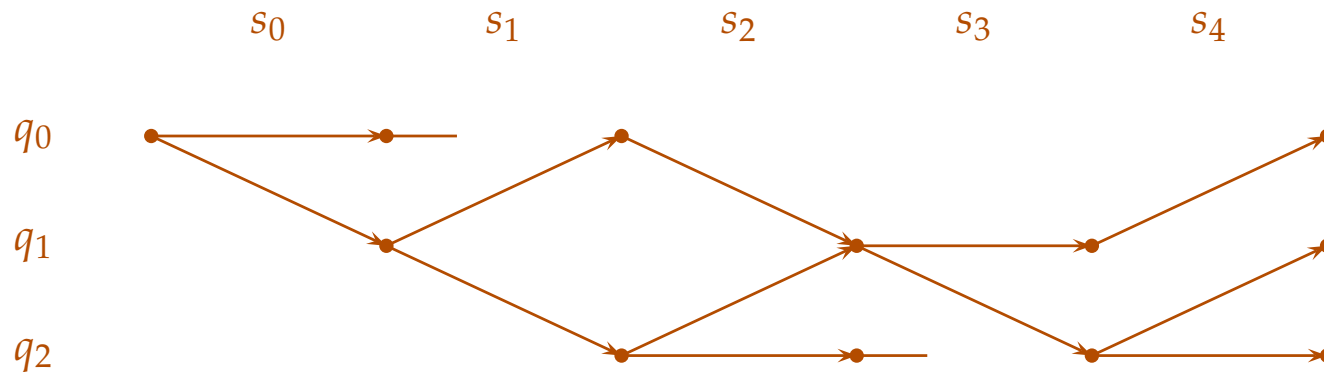
Alternating automata: runs



- run of \mathcal{A} over temporal structure $s_0s_1 \dots$ is a **dag**

initialization dag rooted at initial location q_0

consecution current state and next dag slice satisfy transitions



Alternating automata: runs



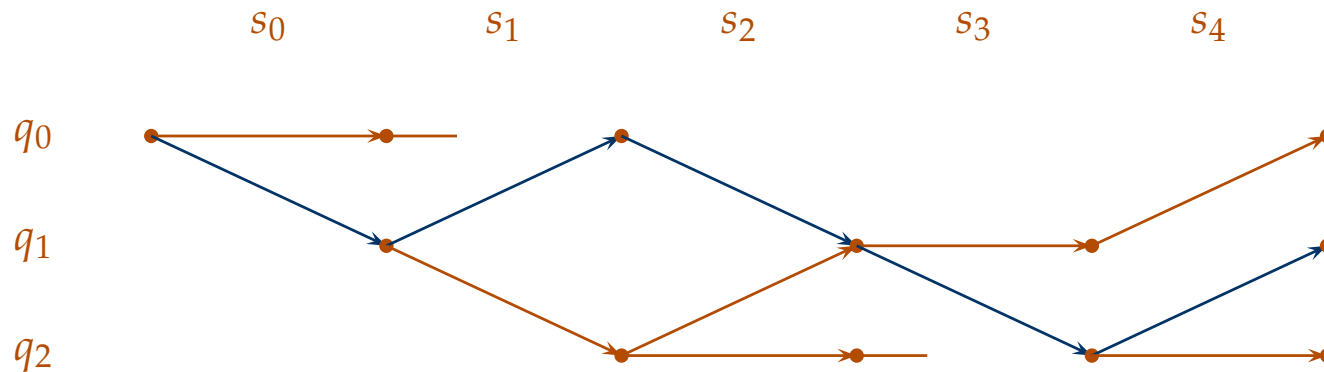
- run of \mathcal{A} over temporal structure $s_0s_1 \dots$ is a **dag**

initialization dag rooted at initial location q_0

consecution current state and next dag slice satisfy transitions

acceptance must hold for **all infinite paths** through the dag

- language: ω -words for which there is accepting dag



Alternating automata: results



- no more powerful than Büchi automata
 - subset construction for translation [Miyano&Hayashi 1984]
 - exponentially more succinct
 - offer both non-determinism and parallelism
- closure properties
 - union and intersection: change initial state
 - complement also simple:
dualize transition relation and modify acceptance condition



Weak alternating automata



• ranking function on states $\rho : Q \rightarrow \mathbb{N}$

$\rho(q') \leq \rho(q)$ whenever q' occurs in $\delta(q)$

“limit rank” every infinite path eventually at constant rank

acceptance limit rank is even



Weak alternating automata

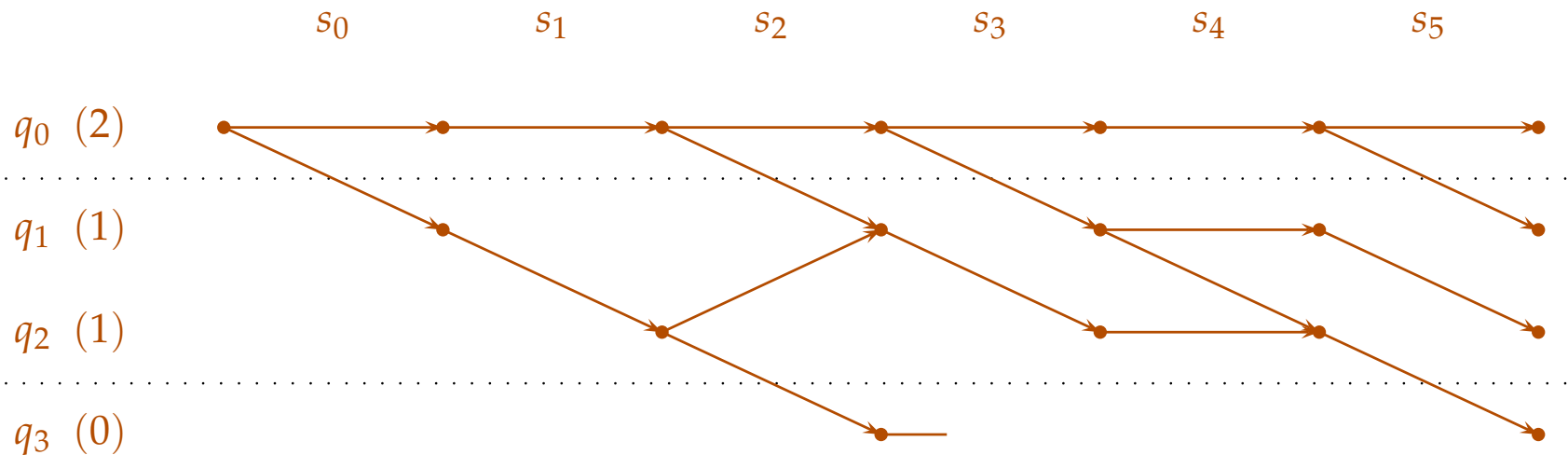


- ranking function on states $\rho : Q \rightarrow \mathbb{N}$

$\rho(q') \leq \rho(q)$ whenever q' occurs in $\delta(q)$

“limit rank” every infinite path eventually at constant rank

acceptance limit rank is even



From LTL to WAA



- construct \mathcal{A}_φ such that $\mathcal{L}(\mathcal{A}_\varphi) = \mathcal{L}(\varphi)$

locations	a location ψ for every subformula ψ
initial location	location φ corresponding to φ
transitions	non-temporal formulas: ensure satisfaction temporal formulas: apply recursion laws
acceptance	ensure odd rank for $\mathbf{F} \psi, \psi_1 \mathbf{U} \psi_2$



From LTL to WAA



- construct \mathcal{A}_φ such that $\mathcal{L}(\mathcal{A}_\varphi) = \mathcal{L}(\varphi)$

locations a location ψ for every subformula ψ

initial location location φ corresponding to φ

transitions non-temporal formulas: ensure satisfaction
 temporal formulas: apply recursion laws

acceptance ensure odd rank for $\mathbf{F}\psi, \psi_1 \mathbf{U} \psi_2$

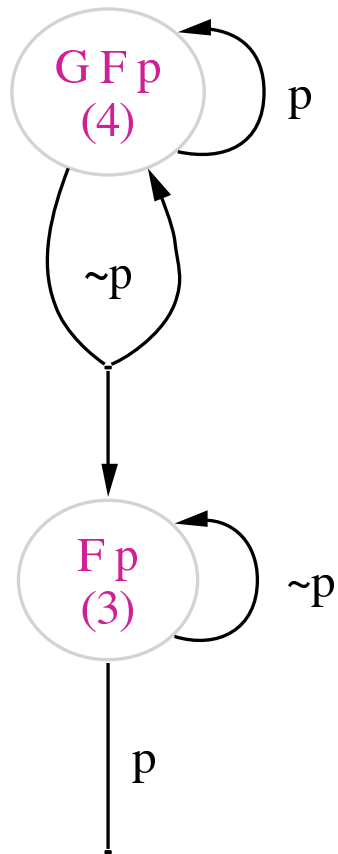
location	δ	ρ
ψ non-temporal	ψ	0
$\mathbf{X}\psi$	ψ	$\rho(\psi)$
$\mathbf{G}\psi$	$\delta(\psi) \wedge \mathbf{G}\psi$	$\geq \rho(\psi)$ even
$\mathbf{F}\psi$	$\delta(\psi) \vee \mathbf{F}\psi$	$\geq \rho(\psi)$ odd
$\psi_1 \hat{\bigvee} \psi_2$	$\delta(\psi_1) \hat{\bigvee} \delta(\psi_2)$	$\max(\rho(\psi_1), \rho(\psi_2))$





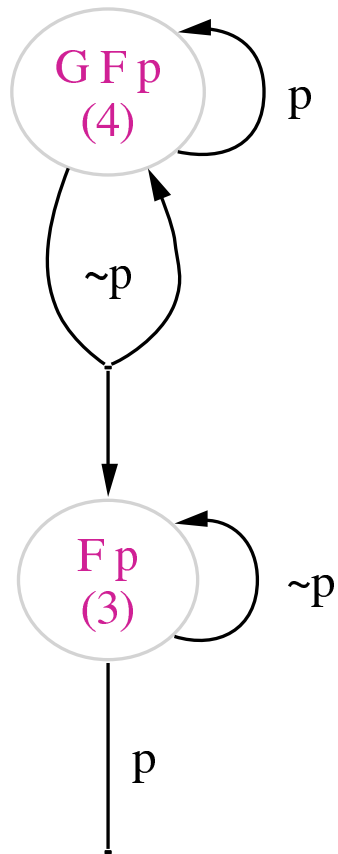
Examples

GFp

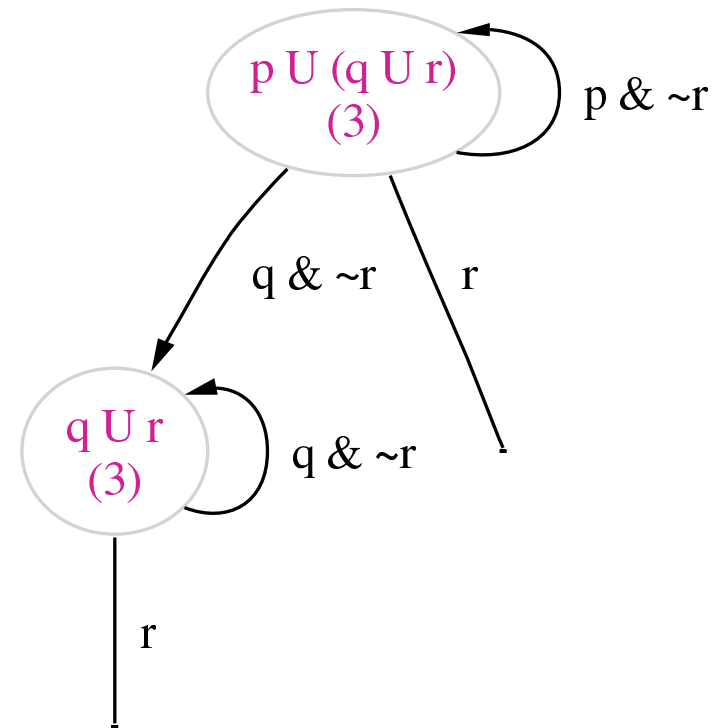


Examples

GFp



$pU(qUr)$



Linear WAA



• format of recursion laws for LTL $\varphi \equiv \dots$

\rightsquigarrow right-hand side: either strict subformulas of φ or $X\varphi$



Linear WAA



- format of recursion laws for LTL $\varphi \equiv \dots$
 - \rightsquigarrow right-hand side: either strict subformulas of φ or $X\varphi$
- **linear WAA:** graph has only trivial loops
 - \rightsquigarrow $q = q'$ holds whenever $q \Longrightarrow^+ q'$ and $q' \Longrightarrow^+ q$



Linear WAA



- format of recursion laws for LTL $\varphi \equiv \dots$
 - \rightsquigarrow right-hand side: either strict subformulas of φ or $X\varphi$
- **linear WAA:** graph has only trivial loops
 - \rightsquigarrow $q = q'$ holds whenever $q \Longrightarrow^+ q'$ and $q' \Longrightarrow^+ q$
- run dag contains no “rising edges”
 - \rightsquigarrow non-accepting infinite path must be trapped at odd state



Emptiness checking



- traditionally: transformation to Büchi automaton

[Gastin&Oddoux 2001, Fritz&Wilke 2003]



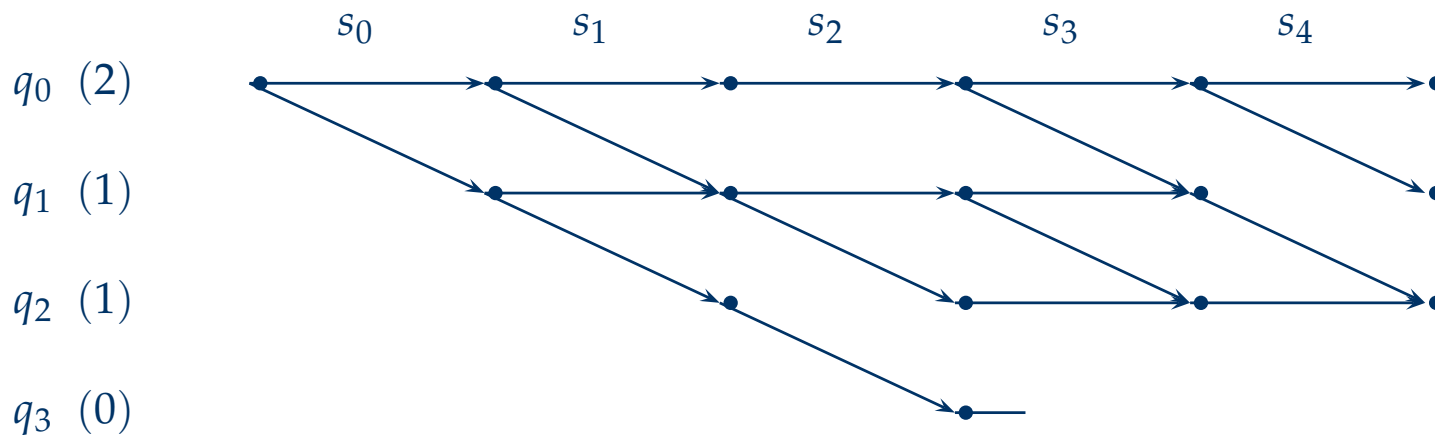
Emptiness checking



- traditionally: transformation to Büchi automaton

[Gastin&Oddoux 2001, Fritz&Wilke 2003]

- direct non-emptiness criterion for linear WAA



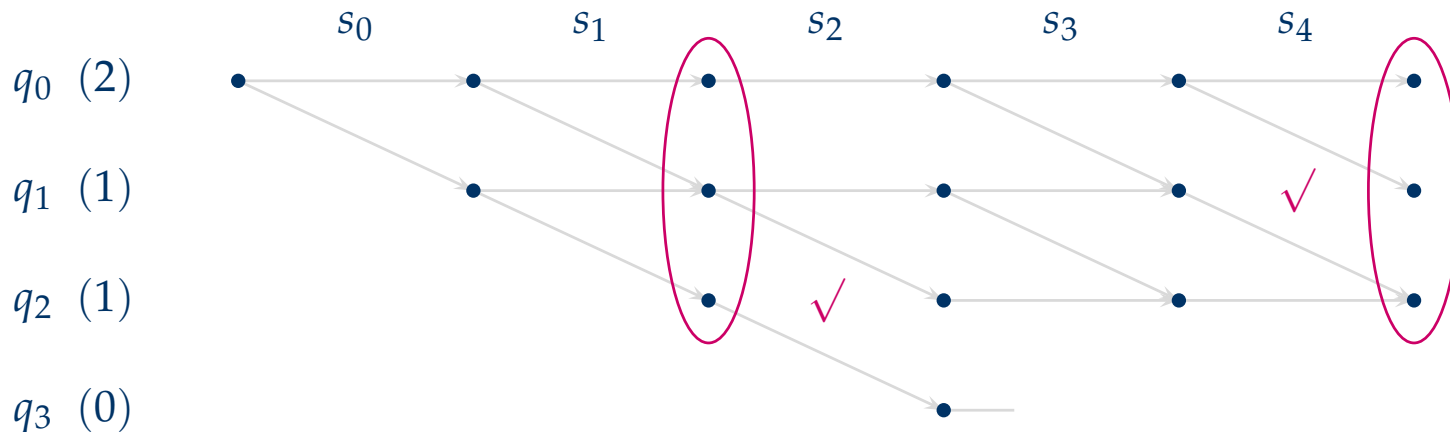
Emptiness checking



- traditionally: transformation to Büchi automaton

[Gastin&Oddoux 2001, Fritz&Wilke 2003]

- direct non-emptiness criterion for linear WAA



$\Rightarrow s_0s_1(s_2s_3s_4)^\omega$ will be accepted by \mathcal{A}



Practical considerations



- prototype implementation
 - encoding of run dags for SAT solvers
 - direct search based on Tarjan's algorithm
- extension to model checking as for Büchi automata
- emptiness checking exponential in size of WAA
 - minimizing WAAs is hard (but automata are small)
 - simple ad-hoc optimizations
 - apply simulation relations

[Fritz&Wilke 2003]



Branching time: principle



- model checking problem $\mathcal{T} \models \varphi$
consider \mathcal{T} as a whole, not its computations



Branching time: principle



- model checking problem $\mathcal{T} \models \varphi$

consider \mathcal{T} as a whole, not its computations

- automata-theoretic approach

[Kupferman et al, 1999]

- alternating tree automaton $\mathcal{A}_{\mathcal{D},\varphi}$

\rightsquigarrow recognize the models of φ with out-degrees in \mathcal{D}



Branching time: principle



- model checking problem $\mathcal{T} \models \varphi$

consider \mathcal{T} as a whole, not its computations

- automata-theoretic approach

[Kupferman et al, 1999]

- alternating tree automaton $\mathcal{A}_{\mathcal{D},\varphi}$

\rightsquigarrow recognize the models of φ with out-degrees in \mathcal{D}

- alternating **1-letter word** automaton $\mathcal{A}_{\mathcal{T},\varphi} = \mathcal{T} \times \mathcal{A}_{\mathcal{D},\varphi}$

\rightsquigarrow simulate run of \mathcal{T} over $\mathcal{A}_{\mathcal{D},\varphi}$



Branching time: principle



- model checking problem $\mathcal{T} \models \varphi$

consider \mathcal{T} as a whole, not its computations

- automata-theoretic approach

[Kupferman et al, 1999]

- alternating tree automaton $\mathcal{A}_{\mathcal{D},\varphi}$

\rightsquigarrow recognize the models of φ with out-degrees in \mathcal{D}

- alternating **1-letter word** automaton $\mathcal{A}_{\mathcal{T},\varphi} = \mathcal{T} \times \mathcal{A}_{\mathcal{D},\varphi}$

\rightsquigarrow simulate run of \mathcal{T} over $\mathcal{A}_{\mathcal{D},\varphi}$

- either $\mathcal{L}(\mathcal{A}_{\mathcal{T},\varphi}) = \{\mathcal{T}\}$ or $\mathcal{L}(\mathcal{A}_{\mathcal{T},\varphi}) = \emptyset$

\rightsquigarrow decide emptiness problem for $\mathcal{A}_{\mathcal{T},\varphi}$



Branching time: results



- CTL and alternation-free μ -calculus
 - $\mathcal{A}_{\mathcal{D},\varphi}$ and $\mathcal{A}_{\mathcal{T},\varphi}$ are WAAs
 - size linear in $|\varphi|$ and $|\mathcal{T}|$
 - emptiness problem for 1-letter WAAs linear



Branching time: results



- CTL and alternation-free μ -calculus
 - $\mathcal{A}_{\mathcal{D},\varphi}$ and $\mathcal{A}_{\mathcal{T},\varphi}$ are WAAs
 - size linear in $|\varphi|$ and $|\mathcal{T}|$
 - emptiness problem for 1-letter WAAs linear
- full μ -calculus
 - $\mathcal{A}_{\mathcal{D},\varphi}$ and $\mathcal{A}_{\mathcal{T},\varphi}$ are parity automata
 - size linear in $|\varphi|$ and $|\mathcal{T}|$
 - emptiness problem for 1-letter parity automata in NP



Branching time: results



- CTL and alternation-free μ -calculus
 - $\mathcal{A}_{\mathcal{D},\varphi}$ and $\mathcal{A}_{\mathcal{T},\varphi}$ are WAAs
 - size linear in $|\varphi|$ and $|\mathcal{T}|$
 - emptiness problem for 1-letter WAAs linear
- full μ -calculus
 - $\mathcal{A}_{\mathcal{D},\varphi}$ and $\mathcal{A}_{\mathcal{T},\varphi}$ are parity automata
 - size linear in $|\varphi|$ and $|\mathcal{T}|$
 - emptiness problem for 1-letter parity automata in NP
- matches time complexity, tightens space complexity





Thank you — questions?

